REMARKS

Applicants respectfully request reconsideration and allowance of the present application. By this Amendment, Applicants amend claims 1, 32, 35 and 43. Claims 1-50 and 53-54 will be pending in the application upon entry of this Amendment.

Allowable Subject Matter

Applicants appreciate the Examiner's indication that claims 32-34 are allowed. Upon review of the Examiner's reasons for allowance, Applicants have noted that the invention does not necessarily require each buffer to store the exact same set of instructions, and have amended the claim to ensure that this limitation is not imported into the claim. However, it did not appear that the Examiner was relying on this implicit limitation as a reason for allowance, and so this amendment should not affect the allowability of the claim. Nevertheless, the Examiner is respectfully requested to contact the undersigned to discuss any concerns.

Moreover, in view of the foregoing amendments and following remarks, Applicants respectfully request reconsideration and allowance of the remaining pending claims.

Claim Rejections Under 35 U.S.C. § 103 in view of Mahalingaiah and George

Claims 1-4 and 8-18 stand rejected under 35 USC § 103(a) as being obvious over U.S. Patent No. 5,989,865 to Mahalingaiah ("Mahalingaiah") in view of U.S. Patent No. 4,626,988 to George ("George"). For reasons set forth more fully below, this rejection is respectfully traversed.

Mahalingaiah and George Do Not Suggest a Buffer That Stores Scheduling Information As Required By Amended Independent Claim 1

Amended independent claim 1 requires, *inter alia*, a buffer in a dispatch stage that is adapted to store a set of loop instructions and scheduling information associated with the instructions. Claim 1 has been amended to more particularly require that "the stored scheduling information defines, for each processor cycle in the loop, whether and which of the stored instructions are to be executed by the functional unit in that processor cycle, such that for a first processor execution cycle, the stored scheduling information is used to determine a first one of

the stored instructions to be selected and issued to the functional unit from the buffer, and such that for a second different processor execution cycle, the stored scheduling information is used to determine a second one of the stored instructions to be selected and issued to the functional unit from the buffer."

The Office Action points to Mahalingaiah's MROM Access as corresponding to the claimed buffer, and further alleges that George teaches receiving instructions from a fetch stage that can be stored in the buffer. However, neither Mahalangaiah nor George teaches or suggests a buffer that stores scheduling information as required by claim 1. Accordingly, the alleged combination of Mahalingaiah and George fails to support a prima facie case of obviousness. MPEP 2143.03.

The Office Action relies on column 17, line 66 to column 18, line 16 and Figure 4 as allegedly suggesting the claimed scheduling information. However, the microinstructions released from MROM Access are stored into Reservation Stations 22. The functionality of the Reservation Stations are summarized in column 10, line 63 to column 11, line 11. These instructions are held in the Reservation Stations until their operands have been produced and other constraints have been satisfied. This means that microinstructions released from the MROM can stay in the Reservation Stations for an arbitrary number of cycles that is not under the control of the MROM unit. More importantly, microinstructions released from MROM at the same clock cycle can stay in the Reservation Stations for very different number of cycles, and may not be executed at all. Therefore, the MROM has no control over the instructions to be executed at each clock cycle, thus no way to dictate the scheduling of instructions. This is in contrast to the explicit requirements of amended claim 1, where instructions released from the buffers will be issued to the functional units so as to be executed in a given clock cycle.

Thus, Mahalingaiah and the other cited prior art fail to satisfy the express limitations of amended independent claim 1. Neither Mahilingaiah nor any other cited prior art provides storage of scheduling information to determine a particular loop instruction to be issued to a functional unit for execution in each clock cycle of the loop. This is a major advantage of the invention since no other structures, such as the Reservation Stations required by Mahilingaiah, will be needed beyond the modulo scheduling buffer.

For at least these reasons, amended independent claim 1 patentably defines over Mahalingaiah and the § 102 rejection of claim 1, together with claims 2-4 and 8-18 that depend therefrom, should be withdrawn.

Claims 8, 10, 12 and 14 Further Patentably Define Over Mahalingaiah and George

Claims 8, 10, 12 and 14 depend directly or indirectly from patentable claim 1 and are patentable for at least the reasons claim 1 is patentable. Moreover, claims 8, 10, 12 and 14 further patentably define over the cited prior art.

For example, claim 8 requires control logic that causes instructions to be issued from the buffer to the functional unit. The control logic receives loop parameters including a loop iteration initiation parameter, a loop iteration parameter and a loop cycles parameter, and wherein the control logic includes an iteration initiation interval register for storing the loop iteration initiation parameter, a loop iteration register for storing the loop iteration parameter, and a loop cycles register for storing the loop cycles parameter.

Mahalingaiah does not use or store the loop information required by these claims for selectively causing instructions to be issued from the buffer to the functional unit. As set forth above, Mahalingaiah relies on the Reservation Stations to collectively determine the instructions to be executed in each clock cycle according to the run-time conditions detected by the Reservation Stations (Mahalingaiah column 4, line 63 to column 5, line 11).

In the claimed invention, the stored loop parameters, including the loop cycles register and the loop initiation interval register, together with the stored scheduling information, allows the buffer control logic to efficiently select, in each clock cycle, the instruction(s) that need to be executed in that clock cycle.

Since the scheduling information and loop parameters are already stored and predetermined, there is no need to for Reservation Stations to perform the run-time condition detection and scheduling activities (Mahalingaiah column 4, line 63 to column 5, line 11). This goes deeply into the differences between the approach of the present invention and the prior art. The claimed modulo scheduling buffer design allows a for a predetermined analysis of operand availability and other conditions to arrange, for each clock cycle, the instructions that should be executed by the functional units. The modulo scheduling buffer of the claimed invention has

enough intelligence to faithfully enforce the arrangement in an efficient manner. On the other hand, Mahalingaiah describes MROM as part of a dynamic scheduling processor where the Reservation Station performs the equivalent analysis of operand availability through run-time condition detection and determine the set of instructions that can be executed by the functional units in each clock cycle. The cost for performing such activities at run-time is high in electrical power consumption as well as hardware complexity, making processors that perform dynamic scheduling not suitable for many low-power application areas.

While not specifically alleged here, the Office Action alleges in another rejection that "George has taught receiving . . . the loop parameters from an instruction stream. . . ." Applicants respectfully disagree. George merely calculates a loop size from a branch target address. Nowhere does George suggest loop parameters received from an instruction stream and used as explicitly required by the claims.

Applicants respectfully submit that the scheduling information in the claim limitations is truly missing from all the cited prior art because those skilled in the art never realized that the scheduling arrangements made in advance for loop instructions can be efficiently enforced by a buffer with some simple logic.

For at least the foregoing reasons, claims 8, 10, 12 and 14 further patentably define over Mahalingaiah and George and the § 103 rejection thereof should be withdrawn.

Claims 9, 11 and 13 Further Patentably Define Over Mahalingaiah and George

Claims 9, 11 and 13 depend directly or indirectly from patentable claim 1 and are patentable for at least the reasons claim 1 is patentable. Moreover, claims 9, 11 and 13 further patentably define over the cited prior art.

Claims 9, 11 and 13 require that the stored instructions consist of a set of kernel instructions and to further require control logic in the processor that is "operative so that the functional unit executes a number of loop iterations of the stored kernel set of loop instructions, a prologue set of loop instructions different than the stored kernel set of loop instructions, and an epilogue set of loop instructions different than the stored kernel set of loop instructions based on the stored kernel set of loop instructions and received loop parameters."

As the Examiner recognizes from other rejections, the well-known definition of prolog, kernel, and epilog instructions of a loop is tied to modulo scheduling, more specifically clock cycles involved in the execution of the loop, as addressed more fully above. At each clock cycle in run-time, the exact selection of instructions to be executed by the functional unit depends on whether the clock cycle falls into the prolog, kernel, or epilog phases of the loop execution.

Mahalingaiah does not have such concept. Mahalingaiah simply releases from MROM some instructions for holding in the Reservation Stations. The Reservation Stations will then detect the run-time conditions and release in each clock cycle some of these instructions in a likely very different arrangement to the functional units for execution. Therefore, Mahalingaiah does not teach the concept of prolog, kernel, and epilog because the MROM unit does not have any notion of the scheduling arrangement of the loop instructions. In fact, Mahalingaiah teaches away from having prolog, kernel, and epilog in buffer design because of his reliance on Reservation Stations for dynamic scheduling.

For at least the foregoing reasons, claims 9, 11 and 13 further patentably define over Mahalingaiah and George and the § 103 rejection thereof should be withdrawn.

Claim Rejections Under 35 U.S.C. § 103 In View Of Mahalingaiah and Subramanian

Claims 5-7 stand rejected under 35 USC 103(a) as being unpatentable over Mahalingaiah in view of U.S. Patent No. 5,867,711 to Subramanian et al. ("Subramanian"). For at least the foregoing reasons, this rejection is respectfully traversed.

Claims 5-7 depend from claim 1 and are patentable for at least the reasons claim 1 is patentable. Claims 5-7 further require (with claim 1 shown for clarity):

> 1. . . . a buffer in the dispatch stage coupled to the functional unit adapted to receive from a fetch stage and store a plurality of the instructions before issue to the functional unit and further adapted to store scheduling information associated with the instructions . . .
>
> 5. A processor according to claim 1, wherein the scheduling information comprises a plurality of loop stage bit masks respectively associated with the plurality of instructions.

6. A processor according to claim 3,

wherein the scheduling information comprises a plurality of loop stage bit masks respectively associated with the plurality of instructions, and

wherein the control logic is further adapted to cause the certain one of the instructions to be issued in accordance with the respective one of the loop stage bit masks associated with the certain one of the instructions.

7. A processor according to claim 4,

wherein the scheduling information comprises a plurality of loop stage bit masks respectively associated with the plurality of instructions, and

wherein the control logic is further adapted to cause the certain one of the instructions to be issued in accordance with the respective one of the loop stage bit masks associated with the certain one of the instructions.

The Office Action alleges that Mahalingaiah taught storing the claimed scheduling information in instruction buffers. However, as the above analysis showed, this is an incorrect assumption.

Moreover, while Subramanian teaches time-stamp at compile-time, a person of ordinary skill in the art would not have understood that a hardware loop buffer adapted to store scheduling information and a simple control logic can eliminate the need for explicitly storing the instruction arrangements for prolog, kernel, and epilog phases of the loop. This removes an important deficiency of processors that employ modulo scheduling compilers such as the one described by Subramanian: excessive code expansion needed to explicitly represent the prolog, kernel, and epilog phases of the loop execution in the machine-level program generated by the compiler.

Applicants respectfully submit that only hindsight reconstruction of the invention using Applicants disclosure would have led one skilled in the art to arrive at the scheduling information and control logic of claims 5-7 from Subramainian's teachings.

For at least these reasons, the § 103 rejections of claims 5-7 should be withdrawn.

Claim Rejections Under 35 U.S.C. § 103 In View Of Mahalingaiah, George and Subramanian

Claims 26-31, 35-41 and 43-49 stand rejected under 35 USC 103(a) as being unpatentable over Mahalingaiah in view of George and further in view of Subramanian. For at least the foregoing reasons, this rejection is respectfully traversed.

Claims 26-31 Patentably Define Over Mahalingaiah, George and Subramanian

Similar to claims 5-7, independent claim 26 also requires storing modulo schedule stage identifiers respectively associated with the stored plurality of instructions, which are used to cause associated instructions to be selectively issued to a functional unit. The Office Action relies on Subramanian for meeting this subject matter admittedly missing from Mahalingaiah.

As set forth above, Subramanian only teaches compiler software for generating a modulo-scheduled program from a dependency graph generated from a target program instruction loop. As part of this process, Subramanian maps physical times for instructions in generated graph so as to arrive at a schedule for instructions in the output program. (see col. 10, lines 5-9). It would not suggest hardware that stores modulo schedule stage identifiers associated with instructions that are used to cause the instructions to be selectively issued to a functional unit, as explicitly required claim 26.

For at least these reasons, the § 103 rejection of independent claim 26, together with claims 27-31 that depend therefrom, should be withdrawn.

Independent Claims 35 and 43 Patentably Define Over Mahalingaiah, George and Subramanian

Claims 35 and 43 require:

- Loop parameters are stored "in control logic of the processor";
- The stored kernel set of loop instructions are caused "to be selectively issued to functional units of the processor" in accordance with the stored loop parameters; and
- The selective issuance of the kernel set of instructions is done "so that the functional units of the processor execute the number of iterations of the kernel set of loop instructions, the prologue set of loop instructions and the epilogue set of loop instructions based on the stored loop instructions."

Mahilingaiah admittedly teaches nothing about prologue, kernel and epilogue sets of loop instructions.

As set forth above, Subramanian only teaches compiler software for generating a modulo-scheduled program from a dependency graph generated from a target program instruction loop. As part of this process, Subramanian maps physical times for instructions in generated graph so as to arrive at a schedule for instructions in the output program. (see col. 10, lines 5-9).

Moreover, Subramanian teaches nothing about storing a kernel set of instructions in a processor, and associated loop parameters, and then selectively issuing the stored kernel set of instructions using the loop parameters "so that the functional units of the processor execute the number of iterations of the kernel set of loop instructions, the prologue set of loop instructions and the epilogue set of loop instructions based on the stored loop instructions." Nowhere does Subramanian teach or suggest such a technique. Rather, Subramanian requires explicitly defining the prologue and epilogue sets of instructions separately from the kernel set of instructions. Accordingly, the alleged combination would not have met all the limitations of independent claims 35 and 43.

Still further, the Office Action alleges that "George has taught receiving . . . the loop parameters from an instruction stream. . . ." Applicants respectfully disagree. George merely calculates a loop size from a branch target address. Nowhere does George suggest receiving loop parameters as explicitly required by the claims.

For at least the foregoing reasons, claims 35 and 43, together with claims 36-42 and 44-50 that respectively depend therefrom, patentably define over Mahilingaiah, Subramanian and George and the § 103 rejection thereof should be withdrawn.

Rejection of Claims Under 35 U.S.C. § 103(a) By Mahalingaiah, George, Subramanian and Valluri

Claims 19-21 stand rejected under 35 USC 103(a) as allegedly being unpatentable over Mahalingaiah in view of George as applied to claims 9, 11 and 13, above, and further in view of

Valluri and Govindarajan's "Modulo-Variable Expansion Sensitive Scheduling" published in *High Performance Computing*, 1998 ("Valluri").[1]

Claims 19-21 depend from claims 9, 11 and 13, which have been shown above to patentably define over Mahalingaiah and George, at least because Mahalingaiah and George do not say anything about prologue, epilogue and kernel sets of instructions, much less a technique for causing all different sets of instructions to be executed based on a stored kernel set of instructions and received loop parameters as required by claims 9, 11 and 13. The alleged combination with Subramanian and Valluri would not have cured this deficiency.

Moreover, Valluri is directed to compiler-based scheduling of code, which further teaches away from the invention, which is based in hardware. Accordingly, one skilled in the art would not be led to the hardware-based invention of claims 19-21, even if, *arguendo*, Valluri could be combined with Mahalingaiah and Subramanian.

For at least these reasons, the rejections of claims 19-21 should be withdrawn.

Rejection of Claims Under 35 U.S.C. § 103(a) By Mahalingaiah, George, Subramanian and Mason

Claims 22-25, 34,[2] 42 and 50 stand rejected under 35 USC 103(a) as allegedly being unpatentable over Mahalingaiah in view of Subramanian as applied to claims 3, 8, 11 and 13, above, and further in view of U.S. Patent No. 6,418,489 to Mason et al. ("Mason").[3]

Claims 22-25 depend ultimately from independent claim 1, claim 34 depends from independent claim 32, claim 42 depends from independent claim 35, and claim 50 depends from independent claim 43. These independent claims have been shown above to patentably define

---

[1] Although only specifically based on Mahalingaiah and Valluri, the Office Action also refers to the rejection of claims 9, 11 and 13 (which rejection was based on Mahalingaiah and George), and Subramanian. Accordingly, it is believed that the Office Action relies on all four of these references.

[2] It is believed that the reference to claim 34 was a typographical error, given its earlier indication of being allowed.

[3] Although only specifically based on Mahalingaiah, Subramanian and Mason, the Office Action also refers to the rejection of claims 3, 8, 11 and 13, which was based on George. Accordingly, it is believed that the Office Action relies on all four of these references for this rejection.

over Mahalingaiah and Subramanian. The further alleged combination of these references with Mason would not overcome the shortcomings of Mahalingaiah and Subramanian as discussed above. Accordingly, claims 22-25, 34, 42 and 50 are patentable for at least the reasons claims 1, 32, 35 and 50 are patentable.

Moreover, Mason does not teach the type of loop iteration required in the rejected claims. For example, claim 34 requires the control logic to be "operative to allow interrupts to be handled at the end of a current one of the number of loop iterations [associated with a kernel], and to complete the number of loop iterations after the interrupt is handled." In contrast Mason merely discloses taking interrupts at the end of loop iterations (including epilogue). In modulo scheduled code according to the present invention, there is no natural and clean end of a kernel iteration; all the iterations are overlapped. Accordingly, Mason's interrupts would not meet the limitations explicitly required by the claims.

For the foregoing reasons, claims 22-25, 34, 42 and 50 patentably define over the cited prior art and the § 103 rejection of the claims should be withdrawn.

Rejection of Claims Under 35 U.S.C. § 103(a) By Mahalingaiah, George and Subramanian

Claim 53 stands rejected under 35 USC 103(a) as allegedly being unpatentable over Mahalingaiah in view of Subramanian and George. This rejection is respectfully traversed.

For reasons set forth above in traverse of other rejections, the cited prior art does not suggest the explicit limitations in claim 53 of control logic in the processor that is adapted to:

- receive loop parameters associated with loop instructions issued to the functional unit from a fetch stage;
- cause a kernel set of the loop instructions issued to the functional unit to be stored in the buffer in accordance with the received loop parameters;
- cause the functional unit to execute a number of iterations of the kernel set of the loop instructions based on the received loop parameters;
- cause the functional unit to further execute a prologue set of the loop instructions different from the stored kernel set of instructions based on the stored kernel set of instructions and the received loop parameters; and

- cause the functional unit to further execute an epilogue set of the loop instructions different from the stored kernel set of instructions based on the stored kernel set of instructions and the received loop parameters

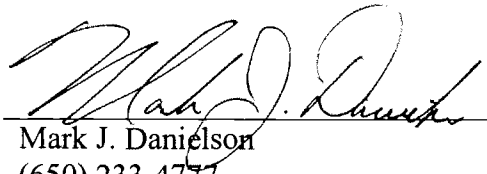
For these reasons, claim 53 patentably defines over the cited prior art and the § 103 rejection of the claims should be withdrawn.

Conclusion

If any issues remain which the Examiner feels may be resolved through a telephone interview, she is kindly requested to contact the undersigned at the telephone number listed below.

Respectfully submitted,
PILLSBURY WINTHROP SHAW PITTMAN LLP

Date: January 17, 2007



40,580
Mark J. Danielson Reg. No.
(650) 233-4777
Please reply to customer no. 27,498